Agent Based Computational Modelling for mapping of Exact $k$Satisfiability representation in Hopfield Neural Network Model

Hamza Abubakar, Sagir Abdu Masanawa, Surajo Yusuf, Yusuf Abdurrahman

Abstract — Recent studies in the field of machine learning and artificial intelligence (AI) are focusing on developing hybrid models to simplify the complexity involved in the training of the neural network. This form of simplicity is valuable for seeking an established convergence artificial neural network. In this paper, agent-based modelling (ABM) using NETLOGO as a platform has been proposed to facilitate the training process of Hopfield neural modelling in carrying Exact $k$Satisfiability programming. The developed ABM hybrid model explored the optimal task representing Exact $k$Satisfiability logic due to the simplicity, flexibility and user-friendly mannerism manifest by ABM model. ABM was used to simulate the process of taking decisions of individual movements, fortification of behaviour, group dynamics, population communications and social interactions within populations. The performance has been displayed based on Global Minimum ratio, local Minimum Ratio, Hamming Distance Mean Square Error and Computation time in evaluating the model performance. The performance of the HN model in carrying Exact $k$Satisfiability (Exact $k$SAT) logic was demonstrated good agreement when compared with ordinary $k$Satisfiability ($k$SAT).

Keywords — Agent base modelling, Artificial Neural network, Hopfield neural network, Satisfiability, Exact Satisfiability, logic program.

1 INTRODUCTION

The field of optimization is one of the important fields to be considered when searching for optimality of decision-making related problem, irrespective of its field (Oulasvirta, et al. 2020)[1]. The topic of decision-making can be illustrated by selecting elite choices in a broad range of choices. It is a rational decision that optimization and decision-makers take to select the most comfortable choices to achieve optimal performance. The area of optimization has recently gained a high degree of publicity due to the tremendous advancement of software, which yields to the achievement of excellent success in the optimization, decision, and search problems. Combinatorial optimization and searching are generally regarded as a subclass of mathematical optimization that has been employed in various field of computational research, whether or not they are simple linear or non-linear or complex in their form (Schulz et al. 2015)[2]. Such mathematical problem can be found everywhere in everyday life, such as the electronic system of vehicles, Scheduling problem, supply chain problem, the fingerprint process used by businesses, the barcode utilized in shopping, the pharmaceutical sector and other systems. The aim of these optimization and search problems is to find the best solution out of varieties of candidate solutions that meet the conditions and specifications of the problem (Aliano et al. 2018)[3]. Artificial neural networks (ANNs) are a part of a family of computational architectural-based models, viewed as equivalent to brain programming by imitating their design and attempting to mimic nervous system activity through which brain information is handled (Yahaya, et al. 2019)[4]. It consists of many basic processing components (called artificial neurons) that are loosely based on biological neurons. This learns about the

---

- Hamza Abubakar is currently pursuing PhD degree program in Mathematical Sciences at Universiti Sains Malaysia, zeeham4u2c@yahoo.com
- Sagir Abdu M. is a Lecturer with the dept. of Mathematical Sciences, Federal University Dutsin-Ma, Nigeria.
- Surajo Yusuf is a Lecturer with Isla Kaita College of Education, Dutsin-Ma, Katsina, Nigeria
- Yusuf Abdurrahman is a Lecturer with Isla Kaita College of Education, Dutsin-Ma, Katsina, Nigeria
algorithm in finding an optimal solution frequency assignment problem in the field of a communications problem. In this work, the main focus is to introduce the optimal capacity of learning in HN model.

2 Agent-based Modelling

The fact that even complicated patterns can be described by computational or analytical models depend on a few fundamental rules of decision is remarkable and such modelling remains a vibrant and crucial research area in an artificial neural network. The core concept behind agent-based modelling is to characterize a structure as a starting point, applying its constituent parts. The "agents" demonstrate the physical properties at their fingertips, and the entire network is then based on the behaviour and interactions of the agents in the system. Thus, the dynamics of this bottom-up method come from a microscale, as opposed to macroscopic simulation models, where complexities are often implemented by equations representing parameters of the model. Agent-based modelling (ABM) applies to the various industrial disciplines in which network can be observed as comprising a complex number of interactive entities ABM has been successfully utilized in modelling dynamics in many areas of research. This include, evacuation models proposed by (Chan et al. 2008)[19], where the network has been describing intuitive as an interactive agent. Every agent's goal is straightforward but interesting consequences may emerge due to their interactions. Agent-based models applied to simulations of traffic operations control (Chen et al. 2010, Bouarfia, et al. 2016, Balmer et al. 2004 and Hofer, et al. 2018)[20, 21, 22, 23]. The objectives of the agents are clear when it comes to controlling traffic operations, but the result of interaction with other agents in the environment, they might work differently from what they would do, they may use all roads on their own, resulting to congestion or even to halt-and-go traffic situation. Grignard et al. (2018)[24] modelled ABM in the city planning field. Where mobile agent models with different needs can provide crucial input on how efficient a specific city plan would be. A new generic algorithm that generates an artificial micro-world which enable the modeller to quickly and easily create and update new mobility scenarios. Various applications of ABM can also be found in social sciences [Davidsson, 2007][25]. Organizational Research (Grand, et al. 2019)[26], tourism (Nicholls, et al. 2017)[27] and the simulation of human systems (Bonabeau, 2002)[28]. ABM can be found in the modelling of ecology and Land Degradation (Alijani, et al. 2017)[29] and complexity research (Mandes et al. 2017)[30]. Electricity markets for sustainable carbon eco-systems. (Babic et al. 2016)[31]. ABM (Kaplar, 2017)[32] has been used to develop an agent-based distribution strategy for an optimal solution to the Travelling Salesman Problem. In more related studies agent-based modelling was developed based on logic programming (Alzaeemi, et al. 2018)[33], Abubakar, et al. 2020)[34], Azim, et al. 2019 and Li, et. 2019)[35, 36]. This can be best understood through models that see this activity not as an external function of the system, but as something that occurs at the agent level and is disseminated across the network. The each ABM can develop based on the following requirements: (i) definition of agents in the environment; (ii) describing the environment; (iii) describing what agents recognize, and what they experience; (iv) describing the goal of the agents in the environment; and (v) Finding rules regarding their behaviour. Although the first four steps are relatively simple for most models, the last step poses some specific challenges. The rules should use the knowledge that agents have access to as input and include what action they take as output, anticipate the minimal rationality (Simon, 1972)[37]. The input that agents are using to make their decisions is easy enough to find and validate. In the evacuation model, for example, dealt with quickly their immediate environment and select their course accordingly, but they have little understanding of the incident behind a physical barrier such as a wall. In the traffic model, agents know which roads are typically congested and which are not but may have access to information on the current construction sites or accidents and base their decisions on this intelligence data. Nonetheless, determining the rules that lead to a conclusion is much more complex and frequently relies on psychological or economic assumptions, which are often difficult to justify, either empirically or theoretically.

A variety of interesting NP-complete issues like Exact kSatisfiability can be closely connected into a set of optimization problems, such as Exact Hitting Set (Carastan-Santos et al. 2017)[38], minimum hitting set(Shi et al. 2010)[39] and Exact cover (Bisoyi et al. 2017)[40], monotone ExactSAT is the same as the exact hitting [Fomin, et al. 2019][41]. set hypergraphs problem onset (Korula, et al. 2009)[42]. Exact graph colouring problem (Guo et al. 2018)[43], exact independent set (Xiao et al. 2017)[44] and it is closely related to the set partitioning problem (Björklund, et al. 2009)[45]. Both problems are NP-complete problems that can be transformed and represented into Exact kSAT and have many applications in combinatorial optimization that falls into the category of optimization, sorting, decision or counting scheme for the solution to the problem of satisfiability is sub-optimal and partially heuristic in nature. However, we are not aware of any previous research work that model the mapping of the Exact kSatisfiability of the Boolean formula in the HN concerning a group of individuals in an agent-based model using the Netlogo framework. We are involved in developing a new approach to modelling NN to be able to catch essential neuron behaviours. The contributions of the present study include the following: (1) To integrate the new logical representation namely Exact kSAT in Hopfield network model (HN); (2) To develop a hybrid agent-based computational model in Hopfield network model (HN); (3) To implement newly proposed hybrid Hopfield network for Exact kSAT based on agent-based computational modelling Netlogo; (4) To explore the feasibility of HN model for Exact kSAT mapping term of a global minimum solution, Local Minimum solution, Hamming Distance, Mean square Error and Computation time; (5) Compare the HN model for Exact kSAT with the existing kSAT model, not recreated, but rather converted into the final published version.

3 Exact kSatisfiability of a boolean formula

The Exact Satisfiability of a Boolean formula is considered as a decision problem which decided a Boolean formula written in Conjunctive Normal form (CNF) has a truth interpretation satisfying exactly one literal in each clause or determine that no such label assignment exists; The Exact kSatisfiability (Exact kSAT) logic is an important variant of Satisfiability (SAT), where the input instance is the same but the question is that in Exact kSAT a clause is satisfied if exactly one of its literals is true (instead of at least one literal, as in ordinary
Exact kSAT. Exact3SAT is also called One-In-Three Satisfiability. Exact kSAT is NP-complete even if subjected to clauses that contain at most three literals and all variables that occur only unnegated (Dahlöf, et al. 2004) [46]. Consider a Boolean expression \( Q_{Exact\, k\, SAT} \) that is built from Boolean variables in CNF which has the following properties:

i. A set of variables \((x_1, x_2, x_3, ..., x_n)\) where \(x_i \in \{1, -1\}\);

ii. A collection of literals. literal represents a boolean variable or its negation \((x_i \text{ or } \neg x_i)\);

iii. A collection of \(m\) distinct logical clauses \(C_i \in \{c_1, c_2, c_3, ..., c_m\}\);

iv. Each satisfying mapping labels exactly one literal in a clause;

v. Each boolean variables in the clause are linked by connectives OR \((\lor)\);

vi. Each clause consists of literal linked by a Boolean connective AND \((\land)\);

vii. Each clause \(C_i\) is a disjunction of exactly three literals.

viii. Each clause \(C_i\) contains at most three literals.

These properties simplify the formulation of Exact kSAT via Hopfield network model (HN) and still preserves NP-completeness (Schaefer, 1978)[47]. The Boolean values for each \(x_i\) are bipolar \(x_i \in \{-1, 1\}\) that exemplifies the notion of FALSE and TRUE respectively. The general formulation of Exact kSAT is presented as follows:

\[
Q_{Exact\, k\, SAT} = \bigwedge_{i=1}^{k} C_i
\]

(1)

when \(k = (1, 2, 3)\) Equation (1) describes the Boolean formula for Exact kSatisfiability containing logical clause \(C_i\) given in Equation (2) as follows:

\[
C_i = \bigvee_{j=1}^{3} (E_{ij}, D_{ij}, F_{ij})
\]

(2)

where \(E_{ij}, D_{ij}, F_{ij} \in \{1, -1\}\). If the context is clear we denote the number of clauses of some formula Exact\(k\)SAT (Porschen et al. 2014)[48]. The Boolean formula for the mapping Exact kSat in HN model is represented in bipolar form. Examples for Exact kSAT formulation \(k = 3\) is presented as follows according to (Schaefer, 1978)[47].

\[
Q_{Exact\, k\, SAT} = (E_1 \lor E_2 \lor E_3) \land (D_1 \lor \neg D_2 \lor D_3) \land (\neg F_1 \lor F_2 \lor F_3)
\]

(3)

Equation (3) is satisfiable since it gives true value resulting \(Q_{Exact\, k\, SAT} = 1\). If the neuron states are considered as \(E_i(i = 1, 2, 3, D_i(i = 1, 2, 3)\) and \(F_i(i = 1, 2, 3)\). The Boolean expression will be unsatisfiable if \(Q_{Exact\, k\, SAT} = -1\). In this study, \(Q_{Exact\, k\, SAT}\) will be embedded in HNN as a proposed model, Exact kSAT-HN model in the next section. The properties of Exact kSAT can be used to govern the behaviour of discrete HN model. In optimization terms, satisfiability problem is classified as a variant of combinatorial optimization in the discrete domain and it first problem to belongs to NP-hard class in most of its variants (Cook, 1971) [49]. A calculus-based search and optimization techniques cannot easily address this type of problem. A strong method is required that can behave in a dynamic way to find an optimal or near-optimal solution to this kind of problem. To the author’s knowledge, the HN model has not yet been used to address Exact Satisfaction Problem in Agent-Based Modelling framework.

4 Mapping Of Exact kSatisfiability in Hopfield Network

The HN model is made up of various components which include the inputs, outputs as well as the synaptic weights. The other two major components of the HN model are its learning mechanism and its energy function. The interconnection of all these functionalities into the HN model makes it an attractive tool for optimization. The basic architecture and structure of the HN model consist of discreet interconnected bipolar neurons with no hidden neurons (Hopfield and Tank 1985)[8]. The synaptic weights are entirely symmetrical, without self-mapping between the interconnected neurons. Hence, the Content Addressable Memory (CAM) of HN model is studied as a dynamical storage network for the synaptic weights (Joya et al. 2002 and Kzar et al. 2016)[9, 17].

Given an initial vector that is mapped to the neuron state \(S^q_i\) where \(i \in N\) and with no noise interference, the HN model will converge to a state of equilibrium corresponding to the constant value of \(H_{min}\) (Barra, 2018 and Sathasivam et al. 2020) [50, 51]. Henceforth, the final configuration of the HN model corresponds to the optimal representation to the combinational problem. Neurons in HN model are represented in a bipolar form \(S^q_i \in \{-1, 1\}\) in conformity to the dynamics of the model structure \(S^q_i \rightarrow \text{sgn}(h_i)\) described by Ising variables in spin glass and the Dean’s problem of mechanical physics (Sherrington, 2010) where the local field, \(h_i\) is described as follows;

\[
h_i(t) = \sum_{j = 1 \text{ or } 1 \neq j}^{N} J_{ij}^q S_j(t) + J_{ij}^q
\]

(4)

The advantage of using bipolar values over binary values is the symmetry of the states of the network. If some pattern \(S^q_i\) in bipolar form is stable, its inverse is stable too whereas. The HN model can be general updated asynchronous obeying the following rule:

\[
S^q_i(t + 1) = \begin{cases} 1, & \text{if } \sum_j J_{ij} S_j(t) + \varphi_j \geq 0 \\ -1, & \text{otherwise} \end{cases}
\]

(5)

where \(J_{ji}\) is the synaptic connection matrix that established the connections between \(j\) to \(k\) neurons, \(S^q_i\) defines the unit condition \(k\) and \(\varphi_j\) describes the threshold function of neurons \(j\). Several studies (Joya et al. 2002, Duong, et al. 2019, Kzar, et al. 2016, Salcedo-Sanz, et al. 2004, Barra et al. 2018, Bag, et al. 2019, Peng, Barra, 2018 and Sathasivam et al. 2020) [9, 15, 16, 17, 18, 50, 51] defined \(\tau = 0\) to verify that the HN model always leads to a decrease in energy monotonically. Each time neuron was connected with \(J_{ji}\) the value of the synaptic connection will be preserved as a stored pattern in an interconnected vector where \(J_{ji}^q = [J_{ji}^{q(1)}]_{1 \times N}, J_{ji}^2 = [J_{ji}^{q(2)}]_{1 \times N}\) and \(J_{ji}^3 = [J_{ji}^{q(3)}]_{1 \times N}\) or \(N\)-dimensional variable vectors \(\tau = (\tau_1, \tau_2, ..., \tau_N)^T\) that the
constraint of synaptic weight matrix \(J^{(1)}\). Self-loop neuron connection is not allowed in HNN model \(J_{ij}^{(0)} = J_{ij}^{(1)},\ldots, = J_{ij}^{(n)} = 0\). It follows the symmetrical neuron synaptic weight matrix \(J_{ij}^{(1)} = J_{ji}^{(1)}\). The HN model energy dynamics function and CAM offers a versatile system with high capacity, error tolerance, rapid memory recovery and partial inputs (Hopfield, 1984, Wen et al. 2009, Fung et al. 2019, Alzaeemi et al. 2018, Barra, 2018 and Sathasivam et al. 2020) [7,8, 12, 15, 33, 50, 51]. To make HN ideal for incorporation with combinatorial optimization such as SAT. HN model can be used as a logical rule to instruct the behaviour of the system configuration based on the synaptic strength matrix. Exact kSAT can be integrated into HN model by allocating the established cost function to each variable with neurons. Besides the generalized cost function \(E_{Q_{ExactSAT}}\) that governs the combinations of HN and \(Q_{ExactSAT}\) is given in eq. (6).

\[
E_{Q_{ExactSAT}} = \sum_{i=1}^{NC} \prod_{j=1}^{m+n} T_{ij}
\]

where \(NC\) and \(m+n\) represent the number of clauses and the number vector variables \(Q_{ExactSAT}\) respectively. Note that, the inconsistency of \(Q_{ExactSAT}\) is given as:

\[
T_i = \begin{cases} 
1 & \text{if } \rho \exists \\
1/2(1-S_i) & \text{otherwise}
\end{cases}
\]

(7)

The real value obtained \(E_{Q_{ExactSAT}}\) is corresponding to the number of “inconsistencies” of the clauses \((E_y = -1, D_y = -1, F_y = -1)\). Minimum \(E_{Q_{ExactSAT}}\) mapping to the “most consistent” selection of \(S_i\). Hence, the updating rule for \(E_{Q_{ExactSAT}}\) in HN model obeys Equation (4) and (5) can be upgraded to embraced third-order connection is defined in equation (8) and (9) respectively as follows.

\[
h_i(t) = \sum_{i=1,r,j,k} \sum_{i=1,r,j,k} J_{ij}^{(3)} S_j(t) + \sum_{i=1,r,j,k} J_{ij}^{(2)} S_j(t) + J_{ij}^{(1)}
\]

(8)

In this case, bipolar values for the output of each neuron are presented as:

\[
S_i(t+1) = \begin{cases} 
1, & \sum_{i=1,r,j,k} \sum_{i=1,r,j,k} J_{ij}^{(3)} S_j(t) + \sum_{i=1,r,j,k} J_{ij}^{(2)} S_j(t) + J_{ij}^{(1)} \geq 0 \\
-1, & \sum_{i=1,r,j,k} \sum_{i=1,r,j,k} J_{ij}^{(3)} S_j(t) + \sum_{i=1,r,j,k} J_{ij}^{(2)} S_j(t) + J_{ij}^{(1)} < 0
\end{cases}
\]

(9)

where \(J_{ij}^{(3)}, J_{ij}^{(2)}, J_{ij}^{(1)}\) are third, second and first order synaptic weights vector of the embedded \(Q_{ExactSAT}\). Equation (8) and Equation (9) is necessary to ensure that neurons \(S_i\) always converge to \(E_{Q_{ExactSAT}} \rightarrow 0\). The Lyapunov energy function \(H_{Q_{ExactSAT}}\) has been utilized to ensures the energy dynamics for the network decrease monotonically. The quality of the retrieved \(S_i\) can be measure by obeying the Lyapunov energy dynamics, \(H_{Q_{ExactSAT}}\), \(k = 2\) defined in Equation (9):

\[
H_{Q_{ExactSAT}} = -\frac{1}{2} \sum_{i=1,r,j,k} \sum_{i=1,r,j,k} J_{ij}^{(3)} S_j S_k - \sum_{i=1,r,j,k} J_{ij}^{(1)} S_j
\]

(10)

The change in energy is always negative until the system reaches its global minimum energy. Equation (10) is a monotonic decrease with the dynamics and can be generalized to include third-order connections as follows:

\[
H_{Q_{ExactSAT}} = -\frac{1}{3} \sum_{i=1,r,j,k} \sum_{i=1,r,j,k} J_{ij}^{(3)} S_j S_k S_l - \frac{1}{2} \sum_{i=1,r,j,k} J_{ij}^{(2)} S_j S_l - \sum_{i=1,r,j,k} J_{ij}^{(1)} S_j
\]

The energy value obtained based on Equation (11) will be classified as the global or local energy level. When the induced neuron configuration has reached global minimum energy, the network will produce the correct solution. Since the energy of the network finitely decreases as the network states change, hence as the number of iterations increases to its maximum level, the energy differential between states is approaching toward zero. Equation (8) demonstrates that the energy depicted from the always diminishes monotonically (Hopfield, 1984)[8]. The value of \(H_{Q_{ExactSAT}}\) obtained refers to the energy value concerning the absolute final energy achieved from \(Q_{ExactSAT}\) (Wan Abdullah, 1992)[52]. As network approaches final energy, the changes in network energy approach zero. Consequently, the quality of the final neuronal state can be fully evaluated obeying the following condition.

\[
|H_{Q_{ExactSAT}} - H_{Q_{ExactSAT}}^m| \leq \xi
\]

(12)

where \(\xi\) is The pre-defined tolerance value of HN model. If the embedded \(Q_{ExactSAT}\) does not satisfy Equation (11), the final state achieved will be stranded in a local minimum solution. It is worth noting that, \(J_{ij}^{(3)} J_{ij}^{(2)} J_{ij}^{(1)}\) can be effectively achieved by using Wan Abdullah’s learning method.

The algorithm of the HN model

Step1: Set \(S_i = (S_1, S_2, S_3, \ldots, S_n)\) as the inputs pattern and store as CAM

Step2: Initialize the weights synaptic matrices, \(J_{ij}^{(3)}, J_{ij}^{(2)}, J_{ij}^{(1)}, J_{ij}^{(0)}, J_{ij}^{(0)}\), and \(J_{ij}^{(1)}\)

Step3: Compute HN model output vectors, \(h_i = (h_1, h_2, h_3, \ldots, h_n)\)

Step4: Compute changes of HN model energy state \(H_{Q_{ExactSAT}}\)

Step5: If the condition stated in equation (12), is not satisfied, go to step 2, and update HN model output vectors.

Step6: If the condition in equation (12) satisfied, proceed to step 7.

Step7: Initialize Exact kSAT logic program and assign neurons to variables.

Step8: If Exact kSAT not satisfied, go to step 2, otherwise proceed to step 9.

Step9: Compute the fitness values of the Exact kSAT using equation (6). If fitness criterion satisfied then end program and print solutions, otherwise, go to step 2.

If the fitness condition is not met, then the program continues iteratively, then it suspends and prints the solutions. The fitness requirement of the HN model is met if the network output condition is satisfied (halts) to some constant value
(which implies that no further optimization emerges in the objective function), no constraints are violated and all the decision variables are non-negative.

5 Learning Phase in Hopfield Network Exact kSAT is Satisfiability

The mathematical formulation of the Exact kSAT logic representation is identical to the Lyapunov energy function of the HN model; hence, the proposed HN model can be used to explicitly address the Exact kSAT problem. Training the HN model minimizes the energy of configuration that the HN model network should "recognize". This allows the network to end up serving as a CAM system the network will converge to a "recollected" state if it is given only part of the state. For instance, if we train a five-unit HN model so that the state (1, -1, 1, -1, 1) is minimum energy, and we assign the network the state (1, -1, -1, -1, 1) it will converge to (1, -1, 1, -1, 1). Consequently, the network is adequately trained when the energy of states the network will consider is local minima. The learning phase of HN model is implemented by incorporating the "behaviour" of Exact kSAT in HN model. It is interesting to note, that the activation of Exact kSAT can be generalized by determining the appropriate synaptic weight of Exact kSAT logic. Exact kSAT logical rule and HN hybridisation can be abbreviated as HN-Exact kSAT model. In HNN-Exact kSAT, deciding the correct synaptic weight for Exact kSAT clause is a primary objective of the learning process. Wan Abdullahi (1992)[52] has pioneered the cutting-edge logic programming model in HN. The clauses given in a logic program need to be transformed into boolean algebra form, based on the Wan Abdullahi method. Then, each of the respective ground neurons is allocated one neuron. In this paper, the implementation of $\text{Q}_{\text{ExactkSAT}}$ in HN is denoted as HN-Exact kSAT model. However, there is no effort in the mapping of Exact kSAT in HN.

Consider the following Exact kSAT program:

$$\begin{align*}
Q_{\text{ExactkSAT}} &= E_1, E_2, E_3 \\
&F_2, F_3 \\
&D_1, D_3 \\
\leftarrow Q
\end{align*}$$

(13)

Given the goal of the program

$$\leftarrow Q$$

where $\leftarrow$ is the conjunction of clauses which defined the goal of the logic program in the HN model. The task of the logic program is to show that $Q_{\text{ExactkSAT}} \leftarrow Q$ is inconsistent to prove the goal $Q$. Sathasivam (2011) based on WA method, any logic program with a set of goals requires to find a consistent interpretation which builds up the model for any given logic program that yields $Q$ true to Boolean representation. If the program label the value 1 to be true and -1 to be the false labelling, then $-Q = -1$ implies a consistent interpretation and $-Q = 1$ indicates at least one of the clause is not satisfied. This can be represented as a combinatorial optimization problem where the "inconsistency" of Equation (13) is to be optimized (Minimization problem). The inconsistency is presented as a negation of Equation (13) after translating all clauses in Equation (13) into Boolean algebraic form; as follows,

$$
\neg Q_{\text{ExactkSAT}} = \neg (E_1 \lor \neg E_2 \lor \neg E_3) \land \neg (D_1 \lor D_3 \lor \neg D_1) \\
\land (F_1 \lor \neg F_2 \lor F_3)
$$

(14)

The costs function of Equation (14) Boolean algebra formula for bipolar representation is that represents $Q_{\text{ExactkSAT}}$ logical inconsistency based on Wan Abdullah learning method is presented as follows:

$$
E_{\text{ExactkSAT}} = \frac{1}{2} (1 - S_{e_1}) (1 - S_{e_2}) (1 - S_{e_3}) + \frac{1}{2} (1 - S_{d_1}) (1 + S_{d_2}) (1 - S_{d_3}) \\
+ \frac{1}{2} (1 + S_{f_1}) (1 - S_{f_2}) (1 - S_{f_3})
$$

(15)

where $S_{e_1}$, $S_{d_1}$, $S_{f_1}$ ($i = 1,2,3$) represent the truth values of neurons $E_1$, $D_1$, $F_1$ respectively. In this work $S_{e_1}$, $S_{d_1}$, & $S_{f_1}$ can take any of the two possible value of 1 (True) and -1(False) for deriving the cost function to be minimized. The optimum value for $E_{\text{ExactkSAT}} \rightarrow 0$, corresponding to the fact that all logical clauses are satisfied. The value $E_{\text{ExactkSAT}}$ is proportional to the number of unsatisfied logical clauses. Since the process involves a massive search space as the number of clause increases, the clause satisfaction is checked systematically by incorporating searching technique such as local searching, exhaustive searching or metaheuristics techniques such as artificial dragonflies. Since the number of correct interpretations will increase proportionally with the number of clauses, a robust training method is required to reduce the complexity (Sathasivam et al. 2010, Mansor et al. 2017, Alzaeemi et al. 2018, Sathasivam, 2020)[53, 54, 33, 50].

The $Q_{\text{ExactkSAT}}$ logic can be regarded as one of the representation problems in the HNN model. This can be implemented in the network by storing atom truth values and generating a optimize cost function when optimal clauses are represented. The value of synaptic strengths can be determined by comparing the cost function and the energy function (Wan Abdullahi, 1992)[52]. The cost function in equation (15), when programmed onto a third-order HNN energy equation in (11) yields the correct synaptic strengths of HNN-ExactSAT as summarized in Table 2, to be stored as Content Addressable Memories (CAM) of HNN. The synaptic weights will, therefore, be used during the recovery phase. The training process provides the optimal cost function to determine the optimal synaptic weights.

Equation (11) can further be expanded and simplified as follows by considering all connections associated with $Q_{\text{ExactkSAT}} (k = 3)$.
In this study, the training process that yields the optimal cost function will determine the best synaptic weights for the system. The computation of optimized global minimum energy involves the correct interpretations and optimized synaptic weights. The optimized global minimum energy can be delineated as the expected global minimum energy to achieve at the end of the retrieval process. The Exact $k$-SAT logic program can be treated as a combinatorial optimization problem. According to Equation (8), $Q_{\text{ExactSAT}}$ consists of three logical clauses with nine variables that are randomly chosen from the set of predetermined variables seven positive literals and two negative literals.

$$S_{E_1} = -1, S_{E_2} = -1, S_{D_1} = 1,$$  
$$S_{D_2} = 1, S_{F_1} = -1, S_{F_2} = 1, S_{F_3} = -1$$

Equation (19) is one of the consistent interpretations that make the entire Boolean formula $Q_{\text{ExactSAT}}$ true. Substituting equation (19) into equation (20), the expected global minimum energy is obtained as follows:

$$H_{\text{QExactSAT}}^\text{min} = -\frac{3}{8}$$

(18)

$H_{\text{QExactSAT}}^\text{min}$ will be used to separate the correctness of the neuron state produced by the network during the retrieval phase.

### 6 Methodology/Implementation procedure

Implementation of Neuro-Heuristic searching method of Exact $k$SAT in HN. The program's main task is to find the best "model" that find the optimal occurrences of Exact $k$SAT Both variables and clauses were initialized randomly. Simulations performed with a different number of neurons complexity ranging from $10 \leq \text{NN} \leq 80$. The executions of these models were carried out on Exact $k$SAT logical representation is presented according to the following steps.

i. Given a logic program:

$$Q_{\text{ExactSAT}} = E_i, E_j, E_k \leftarrow$$

$$F_1, F_2 \leftarrow F_1$$

$$D_1, D_2 \leftarrow D_2$$

ii. Translate all the Exact $k$SAT logical clauses into Boolean algebra form as in Equation (8):

$$Q_{\text{ExactSAT}} = (E_i \vee E_j \vee E_k) \wedge (D_1 \vee \neg D_2 \vee D_3) \wedge (F_1 \vee F_2 \vee F_3)$$

iii. Designate neuron to each variable in an Exact $k$SAT representation.

iv. Randomize the state of the neurons and initialize all connection strengths zero.

$$J_{ik}^{(1)} = J_{ik}^{(3)} = J_{ik}^{(5)} = J_{ik}^{(7)} = J_{ik}^{(9)} = 0$$

v. Derive the cost function, $E_{\text{ExactSAT}}^\text{min}$ for Exact$\text{SAT}$ using Equation (2) - (3).

vi. Equate the cost function Equation (2) and (3) to energy dynamics Equation (6) and obtain the values of the synaptic weight vector $J_{ik}^{(1)}$.

vii. Check clause satisfaction that corresponds to $E_{\text{ExactSAT}}^\text{min} = 0$.

The satisfied assignment will be store as CAM in HNN.

viii. Apply Sathasivam’s relaxation method via equation.

ix. Randomize the state of the neurons. Compute the respective local field $h_i(t)$ of the state space using Equation (4) and (5). If it remains unchanged after five loops, it will be considered as a stable configuration.

Find the corresponding final state of the network $H_{\text{ExactSAT}}^\text{min}$ by using the Lyapunov energy dynamics Equation (6). Check whether the final energy derived is a global or a local minimum based on condition in Equation (7).

Find the terms global minimum ratio ($Z_m$), Local minimum ratio ($Y_m$) and Hamming Distance (HD) and computation time of the to explore the efficiency, accuracy, robustness of implementing the mapping of Exact $k$SAT in HN model.
7 Experimental Setup For HN-Exact kSAT Models

The objective functions for hybridization are for an optimal searching for to Exact kSAT logic representation. The proposed model will be compared with the existing model, HN model kSAT model in the literature. All HN models utilized simulated datasets by generated based on Exact kSAT and kSAT logical clauses. In order to obtain a fair comparison among all HN models, all source code will be implemented based on simulation program developed in ABM with Netlogo Version 6.11 run on a computer equipped with Intel® Celeron® CPU B800@2GHz processor and 4GB RAM using Windows 8. Table 1 until Table 4 display the necessary parameters involved in each HN models.

<table>
<thead>
<tr>
<th>Table 1. List of Parameters and their corresponding values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
</tr>
<tr>
<td>Number of Neurons (NN)</td>
</tr>
<tr>
<td>Neuron Combination (c)</td>
</tr>
<tr>
<td>Tolerance Value (c)</td>
</tr>
<tr>
<td>Selection Rate (η)</td>
</tr>
<tr>
<td>No. Neuron String (μ)</td>
</tr>
</tbody>
</table>

8. Performance Evaluation

The evaluation on performance is a key aspect in the design process of an HN model for exact kSAT and HN model for kSAT. It is deemed that a sufficiently reliable estimate of accuracy and precision of predictions of a model is given by measurements made on the differences between the expected values and those observed, said “difference measures”. Once training process finished, neural network calculated the values for the Global Minimum ratio (Zm), local Minimum Ratio (Ym), Hamming Distance (HD), Mean square Error (MSE) and Computation time (CPU).

\[
Z_m = \frac{1}{n} \sum_{i=1}^{n} H_{Q_{Exact}kSAT} \\
Y_m = \frac{1}{n} \sum_{i=1}^{n} (H_{Q_{Exact}kSAT} - H_{Q_{Exact}kSAT}^{\min}) \\
HD = \frac{1}{tc} \sum_{i=1}^{n} |\Delta H_{Q_{Exact}kSAT} - H_{Q_{Exact}kSAT}| \\
MSE = \frac{1}{n} \sum_{i=1}^{n} (H_{Q_{Exact}kSAT} + H_{Q_{Exact}kSAT}^{\min})^2 \\
CT\_Time = Learning\_Time + Retrieval\_Time
\]

9 Experimental results and Discussions

In this paper, ABM based on Netlogo was used in simulating the results. To validate our proposed network, we generated the simulations result using different number of neurons from 10 ≤ NN ≤ 80. The result of the simulation has been presented in Figure 2-7. The performance of simulated result generated based on HN-ExactkSAT models and existing HN- kSAT model have been presented as follows:

![Figure 2. Global Solution of ABM-HN-Exact kSAT and ABM-HN-kSAT](image)

![Figure 3. Local solution of ABM-HN-Exact kSAT and ABM-HN-kSAT](image)

![Figure 4. Hamming Distance of ABM-HNN-Exact kSAT and ABM-HNN-kSAT](image)

![Figure 5. Mean Square Error of ABM-HNN-Exact kSAT and ABM-HNN-kSAT](image)

![Figure 6. CPU TIME of ABM-HN-Exact kSAT and ABM-HN-kSAT](image)
carrying Exact $k$SAT and $k$SAT logical. It exhibits the behaviours of the Global solution and Local solution, HD, MSE and CPU time observed in the searching processes from $10 \leq NN \leq 80$. In general, both HN model displayed similar searching trend in the behaviours in both cases. When the problem becomes larger, the disparity of HN model searching trend is more noticeable (as the NN increases, both managed to maintain the same NN). The number of satisfied clause decreases sharply as the NN complexity raises). The HN model searching the optimal solution for Exact $k$SAT and $k$SAT logical representation is shown in this study. Figure 2 display $Gm$ achieved conducted via HN models in searching the optimal representation for Exact $k$SAT and $k$SAT logical. The capability of both hybrid networks was measured in term of $Gm$ by considering different NN complexity $10 \leq NN \leq 80$. Sathasivam (2010) and Mansor et al. (2017) [53,45] clarified the connection between the Gm and Lm with the energy nature attained at the end of the simulation time. Hypothetically, if the Gm of the proposed method is approaching one, virtually all solutions in the search space have reached equilibrium globally (100 per cent clauses satisfied). According to Figure 2, HN model based on ABM Netlogo can retrieve a more accurate state for Exact $k$SAT for its ability to hunt closed to 90% success at $NN=80$. HN model manages to get more number of satisfied clauses in Exact $k$SAT than of $k$SAT. Specifically, during $10 \leq NN \leq 20$, HN model achieved Gm 1(100% success) higher than $k$SAT. The real task of the HN model searching is to significantly decrease the learning complexity so that the neurons can proceed to relaxation and recovery process successfully (Sathasivam et al. 2010, Abubakar et al. 2020, Alzaeemi etal. 2018, Sathasivam, 2020)[53, 54, 33, 50]. Furthermore, HN model was able to achieve closed to 90% success in both Exact $k$SAT and $k$SAT searching. It is noticeable that both logic display closed and similar searching trend in term of Gm. In Figure 3 Local solution (Lm) is presented, the lower the Local solution, the higher the acceptability of the searching process of HN model. However, as NN complexity increase Lm will always be going away from zero. Based on Figure 3, it can be observed that the trend displays by HN model were increasing significantly from $30 \leq NN \leq 80$ Exact $k$SAT logic and $20 \leq NN \leq 80$ for $k$SAT logic. Furthermore, HN model recorded 0.875 (87.5%) neurons at local solution for $k$SAT logic which is higher than 0.75 (75%) for Exact $k$SAT logic. Generally speaking, close to 10% of the neuron’s final state obtained both models are in the local solution. Although the complexity of the network is increased by increasing the $NN$ affects the network searching stability and capacity in both cases. Based on the results from Figure 3, local solution portrays in both cases demonstrate a similar trend as the searching complexity increases. This is possible because both case the same only their pattern of satisfiability differed. As a result of the increasing NN, the local solution can also observe to be increasing due to the searching becoming more complicated more local minima will be recorded. The quality of Lm reported in the HN model tends to indicate that it has a substantial capacity to obtain the optimal final state of Exact $k$SAT logic. Figure 5 displays the behaviours of HD for both Exact $k$SAT and $k$SAT logic based on ABM Netlogo. It can be observed that Exact $k$SAT and $k$SAT display lower HD from $NN \leq 20$ and the HD is increasing as the NN increase. It was apparent from the HD trend in Figure 5 that HN model registered HD continuously less than 3% for all $NN$ in Exact $k$SAT logic and increases significantly up to 7% until $NN=80$ for $k$SAT logic. Both models observed to agree with each other in term of HD by manifesting similar movement as demonstrated in Figure 5. However, even as the NN increase, the error value for HD is near to zero in both logics. The solutions obtained are closed to one, so the disparity between the stable states and the attractors is approaching zero. It is back up by having closed to zero HD values. This is because the chance for the same neuron engaged in more than one clause also increased as the number of clauses grew. The lower HD value offers profound proof of HN’s ability to function well with Exact $k$SAT program. In Figures 5, MSE evaluation of HN model for optimal representation of Exact $k$SAT and $k$SAT logic. The magnitude of error in performance evaluation based on MSE in this study is very important. Based on Figure 5, it can be seen that the value of MSE recorded by HN model in searching for Exact $k$SAT and $k$SAT logic representation were increasing significantly from $10 \leq NN \leq 80$. The lower value of MSE manifests the high performance of HN in generating the optimal final state of the model performance. Specifically, during $10 \leq NN \leq 80$, HN model recorded less 7% MSE in Exact $k$SAT lower than $k$SAT logic (8%). It is explicitly shown that both models display similar behaviour in term of MSE. They both show a similar steady rise in errors mapping from $10 \leq NN \leq 80$. However, HN model manifests better performance in searching Exact $k$SAT logic. Accumulation of MSE were observed in both model during the learning stage, as a result, more iteration needed to achieve global convergence in both logic. In this paper, the CPU time can be considered as the time taken for any given program to generate, execute and retrieve the correct Exact $k$SAT and $k$SAT logical representation. Based on Figure 5 CPU Time is required to complete one execution of learning and testing the HN model. According to Figure 5, it can be seen that CPU time increases when the number of crucial points increases. It is seen that in the HN model for ABM with Netlogo consume less time in searching for $k$SAT solution in compares with Exact $k$SAT logic may be due to the nature of searching the correct representation. The different in CPU time between the two logic is significantly large. Moreover, as the searching becomes more complex the $k$ SAT searching show faster convergence compare to Exact $k$SAT logic. Hence, the correct searching of $k$SAT can be obtained in less computation burden for each number than Exact $k$SAT. The lower value of CPU time proves the improved performance of the proposed model in achieving model convergence. Specifically, when the $10 \leq NN \leq 20$, HN model executes $k$SAT logic about 90.5% faster than Exact $k$SAT logic the program. When from $30 \leq NN \leq 80$, the execution of Exact $k$SAT logic is faster than $k$SAT logic the program at about the difference of 93924 seconds. Both models have proven to contend with higher-order neurons, even though the computational cost is higher.

10. Conclusions
We have successfully developed a new logical representation by integrating the Exact $k$SAT in Hopfield network model (HN). The newly proposed hybrid computational mode Hopfield network for Exact $k$SAT was developed based on agent-based computational modelling Netlogo. We have also shown the feasibility of embedded Exact $k$SAT in HN model.
This result showed a giant leap in the implementation of the HN model itself, particularly in terms of producing the correct synaptic weight for effectively connecting the input to output weights with minimal errors. The influence of the optimized output weights produced by the HN model has therefore successfully optimized the training phase with varying neurons levels of complexity. It is possible to attain the ability of the Exact kSAT logical rule on ABM in HN model. Based on this study, we can conclude that the conventional learning approach is not inherently efficient in output weight optimization in ANN. Our proposed model, HN model in Exact kSAT logical representation, has good agreement with the HN model in kSAT logic. The emergence of the Satisfiability Logic Rule in computation is still at an early stage and necessitates more in-depth research, particularly in the other variant of the ANN model, such as the feed-forward neural network. We will be venturing the convolution neural network convolution in the future and optimizing the potential HN model parameters. The approach would potentially open up various possibilities for real-life applications. Additionally, by incorporating robust metaheuristic algorithms such as Dragonfly algorithm, Harmony Search, Bees Algorithm, modified Flower Pollination Algorithm, Ant lion optimizer and particle swarm optimization algorithm and fine-tuning metaheuristics, the HN model learning parameters can be greatly optimized.

References


